Tracker solaire

SAÉ - BUT 3 GEII AII

Della Toffola Antoine - mars 2025

Table des matières :

Introduction

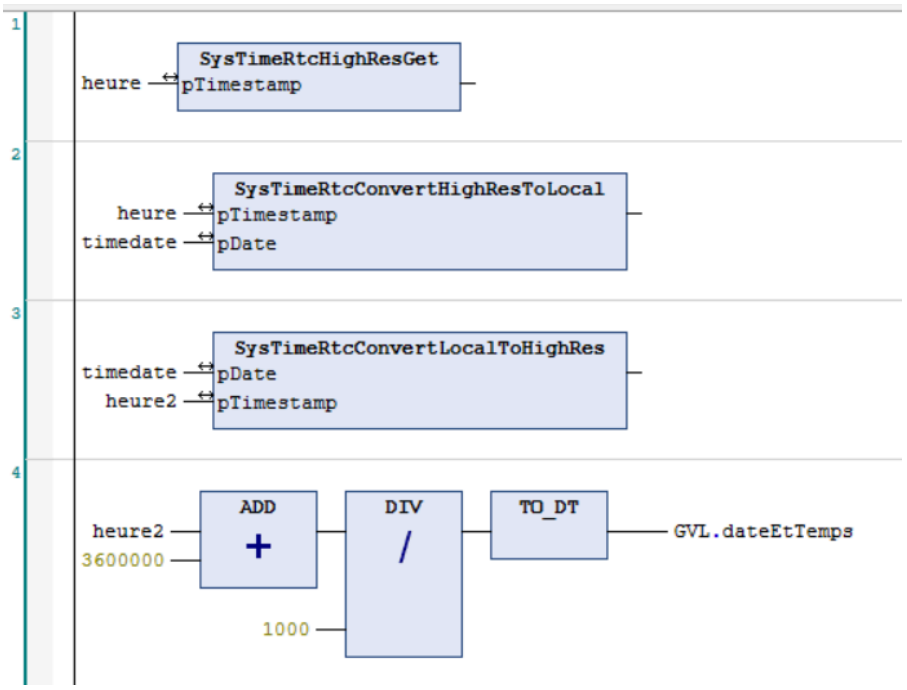
Dans la continuité des précédents rapports liés au fonctionnement du tracker solaire, les valeurs d'élévation et d'azimut du soleil devaient être obtenues via à la station météo intégrant le protocole KNX. Cependant, des problèmes de compatibilité et de liaison entre les équipements (ETS principalement) ont conduit à une alternative consistant à calculer ces variables directement dans l'automate. Pour se faire, un nouveau projet Codesys 3.5 voit le jour liant calculs et commandes des vérins (ce programme à aussi été transmis et expliqué à la partie ombrière, pour le fonctionnement de cette dernière).

Calcul de la position solaire

Les calculs astronomiques de la position du soleil nécessitent des données entrantes que sont la position géographique (longitude/latitude) ainsi que la date et heure au format dateTime.

Acquisition de la date et de l'heure UTC

Pour réaliser les calculs, un programme a été développé pour récupérer et traiter la date et l'heure UTC. Ces valeurs ont été envoyées vers des variables globales afin d'être utilisées dans les calculs ultérieurs.



1

[1] Programme extraction et calcul dateTime automate

Développement des calculs astronomiques

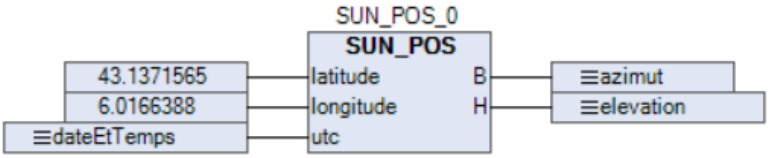
Le calcul de la position du soleil a nécessité l'implémentation d'une série de formules trigonométriques et autres corrections solaire permettant d'obtenir l'élévation et l'azimut. Ces calculs ont inclus plusieurs étapes intermédiaires dont les principales sont :

- Conversion de l'heure UTC en heure solaire locale
- Application des corrections astronomiques
- Calcul de l'angle horaire
- Détermination des coordonnées solaires (azimut et élévation)

```
(* retoune elevation et azymut dans un fonction bloc avec pour entree dateHeure UTC et coordonnees geograpiques *)

n := DWORD_TO_REAL(DT_TO_DWORD(UTC) - 946728000) * 0.000011574074074074;
g :=MODR(6.240040768 + 0.01720197 * n,PI2);
d := MODR(4.89495042 + 0.017202792 * n,PI2) + 0.033423055 * SIN(g) + 0.000349066 * SIN(2.0*g);
e := 0.409087723 - 0.000000006981317008 * n;
cos_d := COS(d);
sin_d := SIN(d);
a := ATAN(COS(e) * sin_d / cos_d);
IF cos_d < 0.0 THEN a := a + PI; END_IF;
c := ASIN(SIN(e) * sin_d);

tau := MODR(0.0174532925199433 * MODR(6.697376 + (n - 0.25) * 0.0657098245037645 + DWORD_TO_REAL(TOD_TO_DWORD(DT_TO_T
rlat := MODR(0.0174532925199433 * latitude, PI2);
sin_lat := SIN(rlat);
cos_lat := COS(rlat);
cos_tau := COS(tau);
tl := cos_tau * sin_lat - TAN(c) * cos_lat;
B := ATAN(SIN(tau) / tl);
IF tl< 0.0 THEN B := B + PI2; ELSE B := B + PI; END_IF;
B := DEG(MODR(B, PI2));
h := DEG(ASIN(COS(C) * cos_tau * cos_lat +SIN(c) * sin_lat));
IF h > 180.0 THEN h := h - 360.0; END_IF;

```

2

Pour organiser ces opérations et faciliter son utilisation par les autres, des blocs fonctionnels ont quasi systématiquement été développés. Ce bloc prend en entrée les coordonnées géographiques et l'heure UTC, et retourne les valeurs d'élévation et d'azimut du soleil.



3

[2] Programme calculs position solaire
[3] Bloc fonction position solaire

Programmes auxiliaires

Afin d'assurer la précision des calculs, plusieurs programmes annexes ont été implémentés, notamment :

- Conversion radian en degré

```
DEG := MODR(57.29577951308232 * RAD, 360.0);
(* conversion en deg *)
```

[4]

- Recréation d'un modulo personnalisé pour certains calculs

```
IF divi = 0.0 THEN
    MODR := 0.0;
ELSE
    MODR := in - DINT_TO_REAL(FLOOR2(in / divi)) * divi;
END_IF;
(* modulo *)
```

[5]

- Correction de valeurs limites pour éviter les erreurs numériques

```
FLOOR2 := REAL_TO_DINT(X);
IF DINT_TO_REAL(FLOOR2) > X THEN
    FLOOR2 := FLOOR2 - 1;
END_IF;
```

[6]

[4] Programme conversion rad vers deg
[5] Programme modulo
[6] Programme correction

Limitation et correction de l'angle d'élévation du soleil

Une fois les calculs astronomiques effectués, un programme de limitation de l'angle d'élévation du soleil a été conçu car l'inclinaison du panneau est contrainte à une plage de [0° ; 45°], correspondant aux limites mécaniques des vérins.

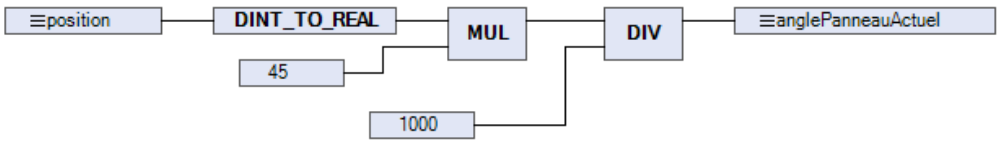
```
(* soustraction de 5 degres a l'elevation car les panneaux a plat ont 5 degres d'inclinaison *)
angleTemp := elevationSoleil - 5.0;

(* limitation entre 0 et 45 degres (limites des verins) *)
IF angleTemp < 0.0 THEN
    GVL.angleFinal := 0.0;
ELSIF angleTemp > 45.0 THEN
    GVL.angleFinal := 45.0;
ELSE
    GVL.angleFinal := angleTemp;
END_IF;
```

[7]

Une correction de -5° a été appliquée, correspondant à l'inclinaison de la structure par rapport au sol que les panneaux conservent lorsque les vérins sont complètement rétractés.

Conversion de la position des vérins en angle

Les vérins étant équipés d'encodeurs, un programme a été développé pour convertir leur position en valeur angulaire. Cette conversion a permis d'obtenir l'angle réel du panneau à partir de la valeur brute renvoyée par les capteurs. Cependant n'ayant pas leur réelle plage de valeur, [0;1000] à été sélectionnée comme valeur de simulation.



[8]

Commande du déplacement des panneaux

Une fois la position réelle du panneau et celle du soleil connues, un programme de comparaison des valeurs a été implémenté pour piloter les vérins.

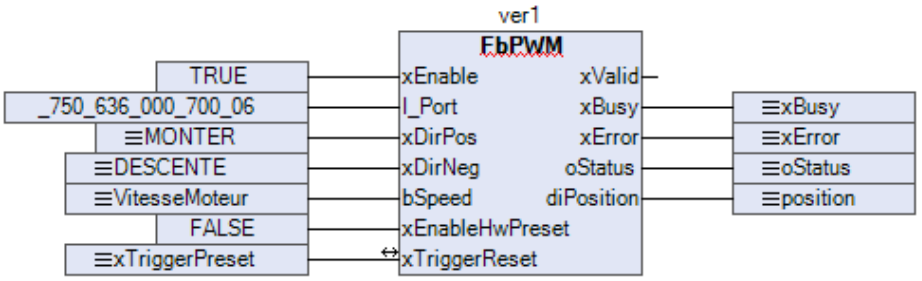
[7] Programme limitation angle
[8] Programme conversion course vérins

Si l'angle du panneau est inférieur à l'élévation du soleil, la commande MONTER est activée. Sinon, si l'angle du panneau est supérieur à l'élévation du soleil, la commande DESCENTE est activée.

```
(* init a false par defaut *)
GVL.MONTER := FALSE;
GVL.DESCENTE := FALSE;

IF AnglePanneau < ElevationFinale THEN
    GVL.MONTER := TRUE;    (* active la montee du panneau *)
ELSIF AnglePanneau > ElevationFinale THEN
    GVL.DESCENTE := TRUE; (* active la descente du panneau *)
END_IF;
```

[9]

Variables qui sont utilisées pour le fonctionnement direct des vérins.



[10]

Conclusion

Le développement du système de tracker solaire aura finalement nécessité de passer outre la station météo, qui, ne voulant coopérer après plusieurs mois de lutte, a dû être remplacée par plusieurs programmes. Allant de la récupération des données temporelles à l'implémentation de calculs astronomiques, en passant par la conversion des données mécaniques et le pilotage des vérins. L'ensemble de ces éléments est intégrable dans l'automate (et déjà intégré dans d'autres parties), garantissant un suivi de la position du soleil à 2 degrés d'erreur près.

[9] Programme commande montée/descente
[10] Commande vérins

Annexes

Bibliothèques CodeSys 3.5

Nom	Localisation	Fonction
WagoAppDCDriveController	addon wago	bibliothèque driver vérins
SysTimeRtc	SysTime	bibliothèque temps automate
SysTypes2	SysTime	bibliothèque temps automate

Tableau de variables

Nom	Type	Localisation	Fonction
temps_0	function block	PLC_PRG	appel de fonction
SUN_POS_0	function block	PLC_PRG	appel de fonction
ver1	function block	PLC_PRG	appel de fonction
limiteAnglePanneau_0	function block	PLC_PRG	appel de fonction
AjustPosition_0	function block	PLC_PRG	appel de fonction
ElevationFinale	REAL	AjustPosition	input de comparaison elevation solaire/angle panneau
AnglePanneau	REAL	AjustPosition	input de comparaison elevation solaire/angle panneau
rad	REAL	DEG	input de conversion en degrés
X	REAL	FLOOR2	input correction
elevationSoleil	REAL	limiteAnglePanneau	input élevation soleil brute
angleTemp	REAL	limiteAnglePanneau	élevation solaire -5°
IN	REAL	MODR	input calcul modulo
DIVI	REAL	MODR	input calcul modulo
latitude	REAL	SUN_POS	input variables calcul
longitude	REAL	SUN_POS	input variables calcul

utc	DateTime	SUN_POS	input variables calcul
B	REAL	SUN_POS	output azimut
H	REAL	SUN_POS	output elevation
g	REAL	SUN_POS	var temporaire calculs trigo
a	REAL	SUN_POS	var temporaire calculs trigo
d	REAL	SUN_POS	var temporaire calculs trigo
t1	REAL	SUN_POS	var temporaire calculs trigo
n	REAL	SUN_POS	var temporaire calculs trigo
e	REAL	SUN_POS	var temporaire calculs trigo
c	REAL	SUN_POS	var temporaire calculs trigo
tau	REAL	SUN_POS	var temporaire calculs trigo
sin_d	REAL	SUN_POS	var temporaire calculs trigo
rlat	REAL	SUN_POS	var temporaire calculs trigo
sin_lat	REAL	SUN_POS	var temporaire calculs trigo
cos_lat	REAL	SUN_POS	var temporaire calculs trigo
cos_tau	REAL	SUN_POS	var temporaire calculs trigo
cos_d	REAL	SUN_POS	var temporaire calculs trigo
rad	REAL	SUN_POS	var temporaire calculs trigo
PI	REAL	SUN_POS	pi
PI2	REAL	SUN_POS	pi²
heure	SYSTIME	temps	var calcul timeDate
timedate	SYSTIMEDATE	temps	var calcul timeDate
heure2	SYSTIME	temps	var calcul timeDate
dateEtTemps	DATE_AND_TIME	GVL (var global)	var dateTime automate
azimut	REAL	GVL (var global)	var azimut

elevation	REAL	GVL (var global)	var elevation
MONTER	BOOL	GVL (var global)	commande vérin
DESCENTE	BOOL	GVL (var global)	commande vérin
VitesseMoteur	BYTE	GVL (var global)	var vitesse vérin
xTriggerPreset	BOOL	GVL (var global)	var reset erreur vérin
xBusy	BOOL	GVL (var global)	traitement info vérin
xError	BOOL	GVL (var global)	erreur vérin
ostatus	WagoSysErrorBase.FbResult	GVL (var global)	statut erreur vérin
position	DINT	GVL (var global)	var position vérin
angleFinal	REAL	GVL (var global)	elevation limitée/corrigée
anglePanneauActuel	REAL	GVL (var global)	position panneau